CS 7641 Machine Learning

Assignment 3: Unsupervised Learning and Dimensionality Reduction

Yurong Fan

Abstract

The organization of this article is as following – in Part 1, unsupervised clustering algorithms (k-means and expectation maximization) are used to cluster two datasets of interests; In Part 2, the two datasets were projected into lower dimensions with the help of 4 dimension reduction algorithms (principal component analysis, independent component analysis, randomized projection, and factor analysis); Part 3 to 5 are conducted to see how dimension reduction can be used together with unsupervised (clustering) and supervised (neural network) learning to reduce computational complexity and in the meantime retain accuracy.

Introduction of the datasets

Dataset 1- Digit Recognition: the digit recognition is one of the most classic computer vision problems. The features in this dataset are 784 pixel values of handwritten images of 10 digits (10 classes for classification). I took a 30% random sample from the original dataset (can be downloaded from https://www.kaggle.com/c/digit-recognizer).

Dataset 2 – Cover Type: this problem is predicting forest cover type from features such as elevation, aspect, slope, distance to hydrology, distance to roadways, and soil type. There are 54 attributes (one-hot-coded categorical variables and numerical attributes), and 7 classes in the dataset. I took a 10% random sample from the original dataset (can be downloaded from https://archive.ics.uci.edu/ml/datasets/Covertype).

The actual number of instances and distribution of classes after sampling are as below.

Dataset 1 – Digit Recognition			Dataset 2 – Cover Type		
Class	#rows	%rows	Class	#rows	%rows
1	1429	11%	2	28440	49%
7	1334	11%	1	21072	36%
3	1298	10%	3	3567	6%
2	1276	10%	7	2051	4%
6	1243	10%	6	1749	3%
0	1242	10%	5	925	2%
4	1236	10%	4	297	1%
9	1219	10%			
8	1204	10%			
5	1119	9%			

Figure 1 - Number and distribution of instances by class of Dataset 1 and Dataset 2

Part 1. Clustering

In part 1, I applied two unsupervised clustering methods with different number of clusters on both datasets, used several evaluation metrics to select the best number of clusters. And see if the numbers of clusters suggested by this process align with the numbers of classes in the supervised setting.

K-means clustering (KM)

K-means algorithm minimize the sum of distances between each data point and its closest cluster centroid by iterating over two steps until converge: 1) assigning each training example to the closest cluster centroid, and 2) moving each cluster centroid to the mean of the points assigned to it. As k-means relies on calculating the distance between points, specifically the Euclidian distance in the classic k-means, attributes of larger scale will dominate the variance in the distance. Thus I used standard scaler (an alternative is the min-max scaler) to bring all attributes to similar scales before clustering.

To determine the optimal number of "k", I ran k-means for a range of values of k, collected clustering evaluation metrics for each k, and plotted curves in Figure 2 to 5. The explanation of the metrics in my setting are as below.

- Adjusted rand index: computes similarity between clustering and class labels adjusted for chance.
- Normalized/adjusted mutual information score: mutual information is a measure of the mutual dependence between the clustering and class labels.
- Homogeneity: equals 1 if all of the members within the same cluster has the same class.
- Completeness: equals 1 if all members of the same class are in the same cluster.
- V-measure: is the harmonic mean between homogeneity and completeness.
- Silhouette: measures how close each point in one cluster is to points in the neighboring clusters.

Metrics excepting silhouette in the first three sub-plots compare the clustering with a ground truth -- in my case the class labels. Their suggested optimal k (6 – 12) are generally more close to the number of classes (10) for Dataset 1 from Figure 2. Based on homogeneity and completeness, when k = 10, around 40% examples within the same cluster has the same class, and around 43% examples of the same class are in the same cluster. It indicates a good alignment between the clustering and class labels. And I think in a perfect situation when examples of the same class are close to each other while far away from examples of a different class in the Euclidian space, choosing k equals the number of class labels will give a clustering exactly match the class labels. However, it's still far from perfect alignment because 1) clustering has no feature selection and all attributes contribute equally which is not ideal as there might be irrelevant or less important attributes, 2) the decision boundaries of different classes can be complex and hard to be represented in the Euclidean space without data transformation, 3) "curse of dimension" will be an issue if the number of attributes is too high for the number of examples.

Silhouette which doesn't require ground truth, suggest k=2, and has close to 0 values when k is around 10 indicating examples are on or very close to the decision boundary between two neighboring clusters. I think it's because examples of the same class are not grouped as "clouds" in the Euclidean space and allow some space from other "clouds". If we don't know the number of classes, silhouette will fail to guide us toward the number of classes in dataset 1 for the similar reasons discussed in the above paragraph when we select k equals the number of classes but still cannot fully align the cluster assignment and the class labels.

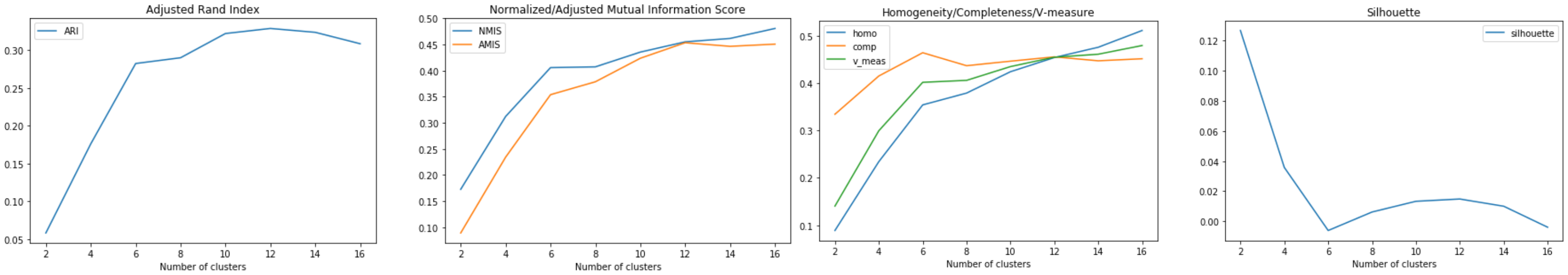


Figure 2 - Applying k-means Clustering on Dataset 1

Similarly, ground-truth based metrics suggested k = 6 which is very close to the number of classes (7). The missed 1 cluster is probably due to the class which only makes 1% of all examples. Silhouette again failed to suggest a k close to the number of classes with reasons discussed above.

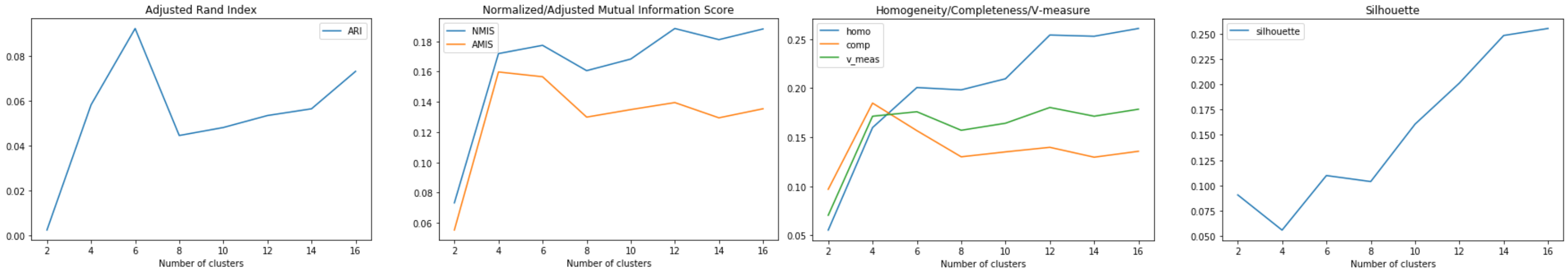


Figure 3 - Applying k-means Clustering on Dataset 2

Expectation Maximization (EM)

The EM algorithm is used to generate the best hypothesis (maximum likelihood hypothesis) for the distributional parameters of some multi-modal data. In clustering, each data points are assumed to be generated from a Gaussian distribution, and each Gaussian distribution represents a cluster. EM algorithm iterates over 2 steps until converge to find the optimal cluster assignments: 1) E-step performs probabilistic assignments of each data point to clusters based on the current cluster hypothesis; 2) M-step updates the cluster hypothesis based on the new data assignments. It extends the k-means clustering as it allows “soft” cluster assignments (each data point can be assigned to multiple clusters with probabilities).

The ground-truth based metrics are generally peaked at 10-12 for dataset 1. Comparing with metrics of k-means which peaked at 6-12, EM with “soft” clusters actually provide more accurate cluster assignments especially for data points far away from cluster centers. For dataset 2, the ground-truth based metrics are peaked at 6 which is the same as the metrics in k-means, and completeness and homogeneity are a little higher at 6 in EM. Same as in k-means, the ground-truth free metric silhouette didn’t suggest a cluster number that is close to the number of class labels.

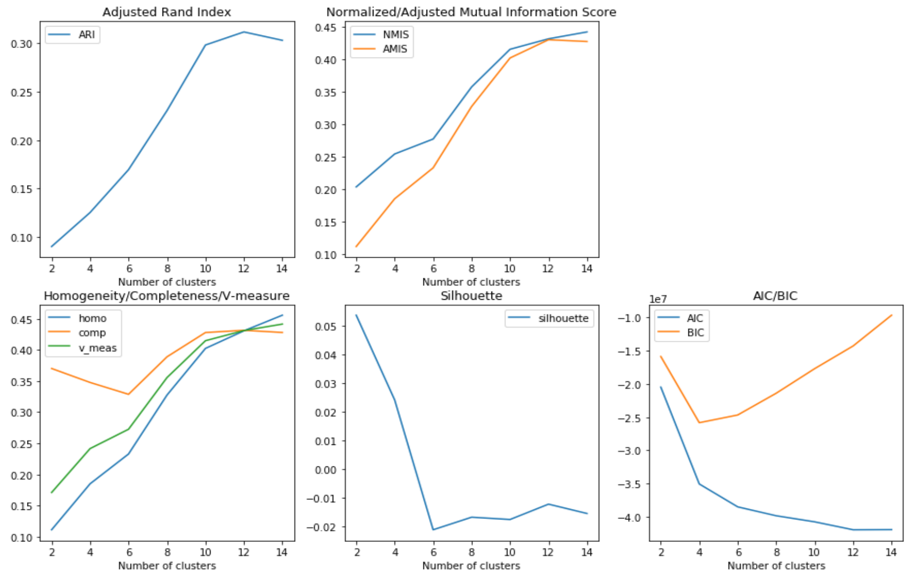


Figure 4 - Applying EM Clustering on Dataset 1

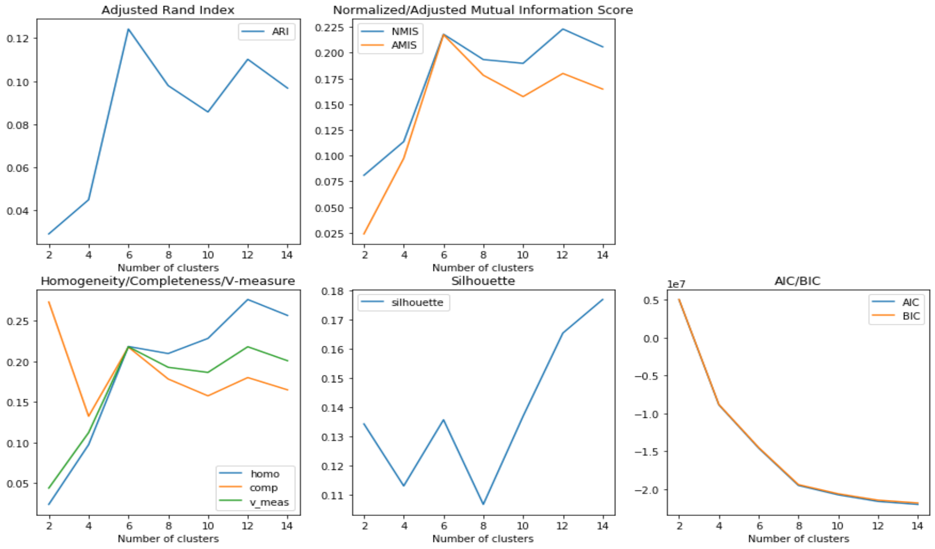


Figure 5 - Applying EM Clustering on Dataset 2

Part 2. Dimensionality reduction

In this part, I did dimensionality reduction of the two datasets with four different algorithms.

Principal Component Analysis (PCA)

PCA is used to decompose a multivariate dataset in a set of successive orthogonal components that explain a maximum amount of the variance. From Figure 6 and Figure 7 for dataset 1 and dataset 2 respectively, PCA identifies the first component with the highest explained variance, the second component with the second highest explained variance and so forth with diminishing variance explained. In dataset 1, PCA used around 300 components to explain near 100% variance in the data of 784 attributes. While in dataset 2, apart from several top components who has relatively high variance explained, majority components that are in the middle of the distribution in Figure 7 have similar medium level variance explained, and PCA needs over 40 components (the data has 54 attributes) to explain near 100% variance.

This observation can be explained by the nature of the attributes in the two datasets. In dataset 1, the attributes are pixel values and are redundant because the values of adjacent pixels in an image are highly correlated. In dataset 2, the attributes are each a real word features such as elevation, aspect, slope, distance and have less redundancy. One learning is that PCA can do a good job reducing the number of features in situations similar to dataset 1, but not in situations similar to dataset 2.

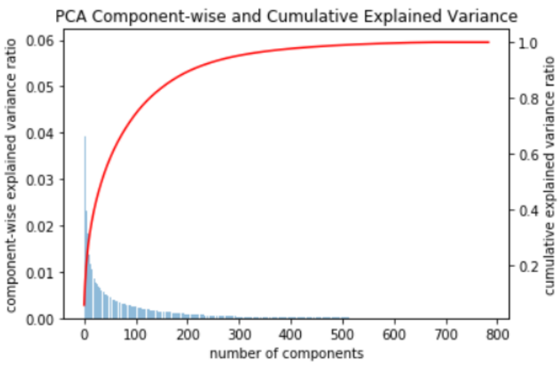


Figure 6 - Applying PCA on Dataset 1

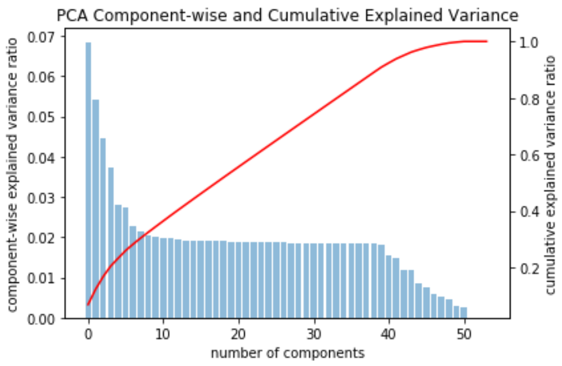
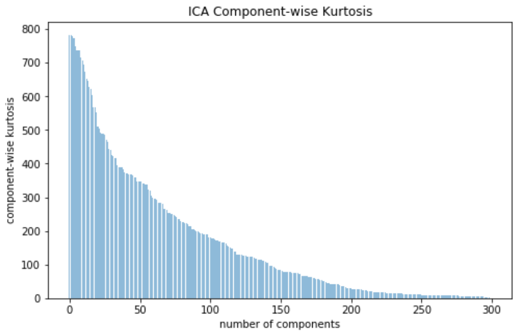


Figure 7 - Applying PCA on Dataset 2

Independent Components Analysis (ICA)

Independent component analysis separates a multivariate signal into additive subcomponents that are maximally independent. It is classically used to separate mixed signals (*blind source separation*). It can also be used as a non-linear decomposition. From central limit theorem, the sum of independent variables follows Gaussian under some conditions. The independent variables need to be non-Gaussian for ICA to recover.

Kurtosis measures the degree of spikiness of a distribution. It is zero for Gaussian, positive if the distribution is spikier than Gaussian, and negative if the distribution is flatter than Gaussian. Figure 8 and Figure 9 are kurtosis distribution of ICA components obtained from dataset 1 and dataset 2 respectively. (I sorted the components by kurtosis). In dataset 1, around 200 ICA components have very non-Gaussian distributions. In dataset 2, around 30 components have non-Gaussian distributions.



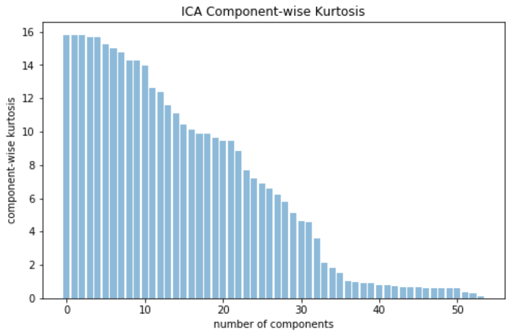


Figure 8 - Applying ICA on Dataset 1	**Figure 9 - Applying ICA on Dataset 2**

In Figure 10, I visualized some components for each algorithm using dataset 1. ICA components look more like meaningful elements that present in the original data. For example, ICA component 2 has a white spot in dark grey background, and component 3 has a black spot in light grey background. I expect getting “bigger” elements when reducing the number of components.

Randomized Projection (RP)

Randomized projection projects data into a lower-dimensional space using a random dimensional matrix whose columns have unit lengths. According to Johnson-Lindenstrauss lemma, random projection can well preserve the distances between points. It is gaining popularity as it can “yield results comparable to conventional dimension reduction methods such as principal component analysis”, and is “computationally significantly less expensive” (Bingham, E. (2001)).

Factor Analysis (FA)

Factor analysis is used to describe variability among observed, correlated variables by searching potentially lower number of unobserved variables called factors. It is closely related to principal component analysis

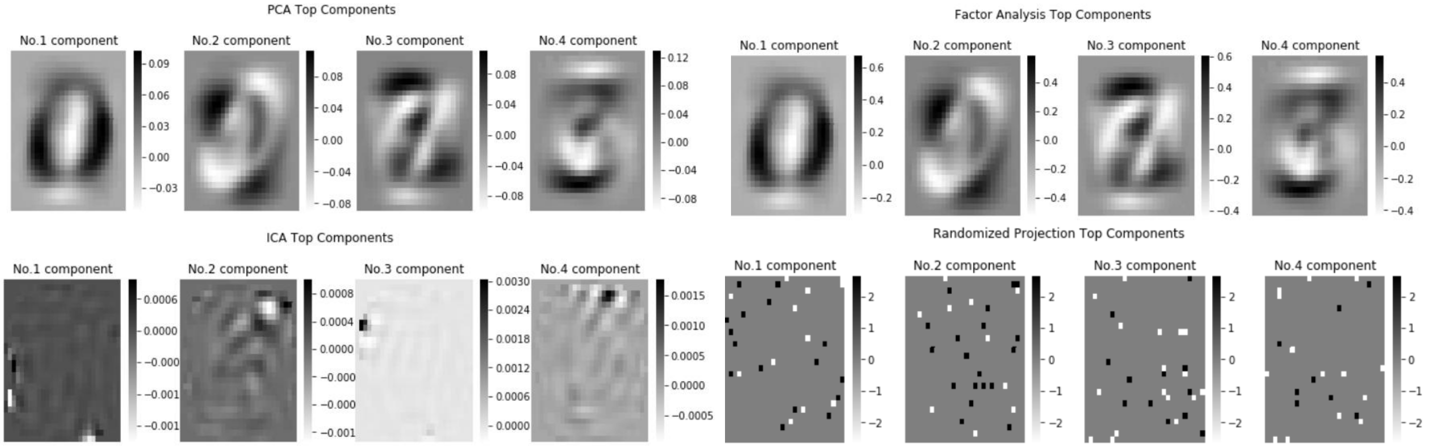


Figure 10 – Visualizing components for all dimension reduction algorithms for Dataset 1

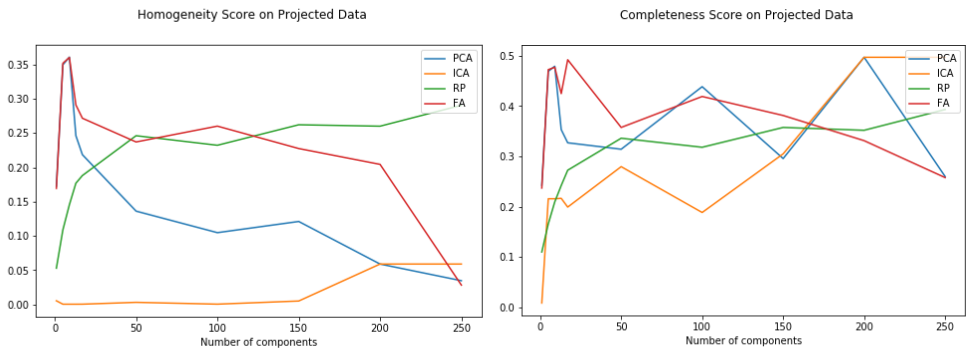
Part 3. Clustering after dimensionality reduction

In this part, I tried to reproduce the clusters generated in Part 1 using datasets projected by dimension reduction algorithms with different number of components. The number of clusters is a fixed value chosen from part 1. Specifically for dataset 1, I used k = 6 for k-means, and number of clusters = 10 for EM; and for dataset 2, k = 6 in k-means, and number of clusters = 6 as well for EM.

Figure 11 and Figure 12 plot performance metrics of k-means and EM respectively on the projected dataset 1. We can see that PCA and FA have the best homogeneity and completeness when having small numbers of components (6 to 10). This can be explained by the connection between PCA and K-means proposed by Ding & He (2004). They proved that principal components are actually the continuous solution of the cluster membership indicators in the k-means, i.e., the PCA automatically performs data clustering according to the objective function of k-means. In Ding & He's experiment, they also applied k-means on PCA projected data, and see significantly improved performance at small level of components.

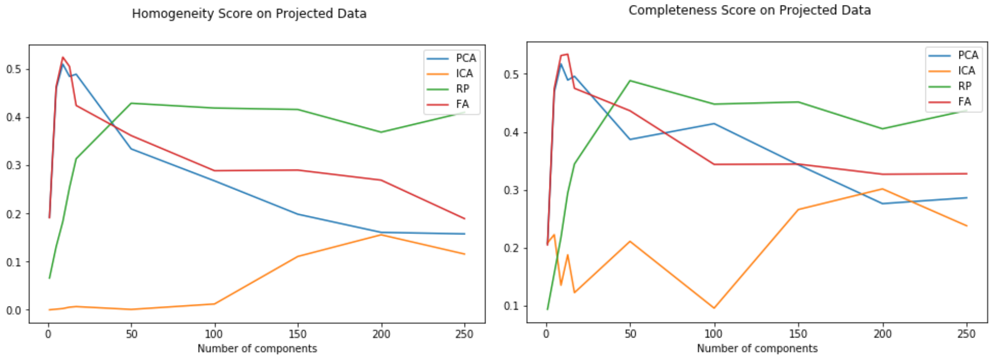
Comparing with the clustering performance on the original dataset 1 in Part 1, the best homogeneity and completeness of clustering on PCA projected data are as good for k-means, and better for EM (e.g. original homogeneity = 0.4, best homogeneity on projected data = 0.5). The performance gain comes from the noise reduction property of PCA.

RP and ICA didn't generate projections that reproduced the clustering in Part 1 regarding the homogeneity and completeness scores. The clustering performances on RP and ICA projected data generally increased as the number of components increased, but they are still way below the original performance when the number of components reached 250. From literatures, Boutsidis, C. (2010), Cohen, M. B. (2015) mathematically proved that random projection can approximate SVD (singular value decomposition) methods such as PCA in dimensionality reduction for k-means clustering with a relative error and computational complexity improvement.


Homogeneity Score on Projected Data

Completeness Score on Projected Data

Figure 11 - Applying k-means on Projected Data for Dataset 1


Homogeneity Score on Projected Data

Completeness Score on Projected Data

Figure 12 - Applying EM on Projected Data for Dataset 1

The same analysis was done for dataset 2 and plotted in Figure 13 and Figure 14. Similar to dataset 1, PCA and FA best reproduced the clustering in Part 1 at small number of components with as good or slightly better homogeneity and completeness. Clustering performance of ICA and RP projection increases with the number of components. But is generally worse than ICA/FA's, excepting that k-means on ICA projection achieved good completeness at high number of components.

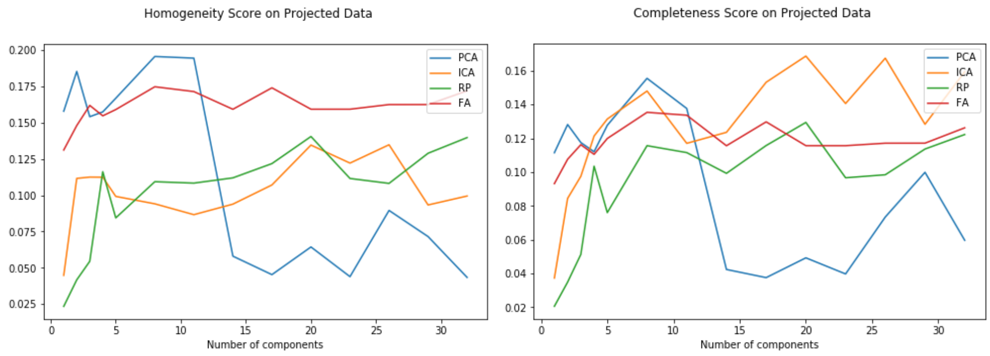


Figure 13 - Applying k-means on Projected Data for Dataset 2

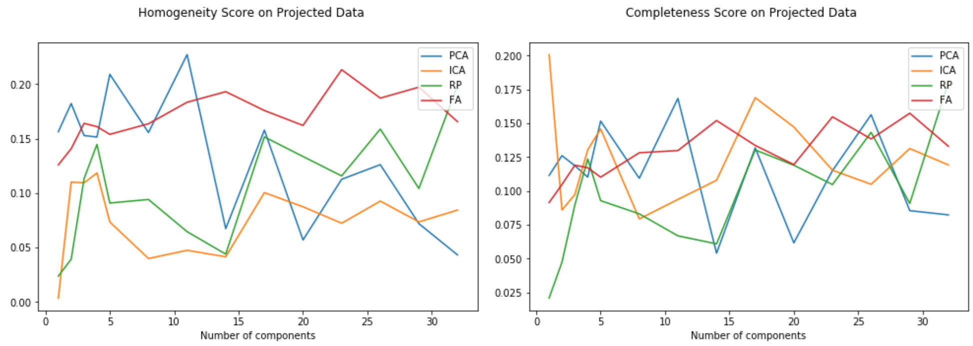


Figure 14 - Applying EM on Projected Data for Dataset 2

Part 4. Neural network learner on dimensionality reduced dataset

This part explores how dimensionality reduction would influence the performance of neural network (NN). A dimension reduction algorithm is applied on the original dataset, and then a neural network is applied on the projected data. The same neural network is applied on the original data with all attributes as a benchmark. Figure 15 shows the F1 score (macro F1 is used for the multi-class classification problem) and average fit time on dataset 1, Figure 16 shows the performance on dataset 2. Performance of neural network on the original data with all attributes is plotted by the dashed lines.

Regarding model accuracy, NN on PCA/FA/RP projected data of both datasets reached the cross validation F1 score of NN on original datasets with number of components much less than the number of original attributes. It indicates that dimension reduced datasets not necessarily worsen neural network accuracy. But rather on my two datasets, dimension reduction improved model accuracy that might due to the reduction of noise, and correlations in the data. It is worth emphasizing that RP achieved comparable NN accuracy as other dimension reduction algorithms. This is consistent with researchers such as Bingham, E. (2001) who conclude RP can yield results comparable to conventional dimension reduction methods such as PCA if the training is based on interpoint distances and RP is not sensitive to impulse noise.

In terms of speed, NN on all 4 types of projected data has much lower average fit time on dataset 1. One reason is the whitening process (de-correlating and normalizing the data to have the same variance) performed by all algorithms. LeCun et al. (1998) argues that the network training converges faster if its inputs are whitened. It is notably that the average fit time doesn't necessarily increase with the number of components as the number of iterations needed to converge doesn't necessarily increase. Bingham, E. (2001) argues that RP is computationally significantly less expensive than other dimension reduction algorithms. From my observation, this efficiency of RP is achieved in the projection phase rather than the neural network fitting phase. It is surprising that NN on projected data has higher fitting time in dataset 2, and its cause can be a topic of future work.

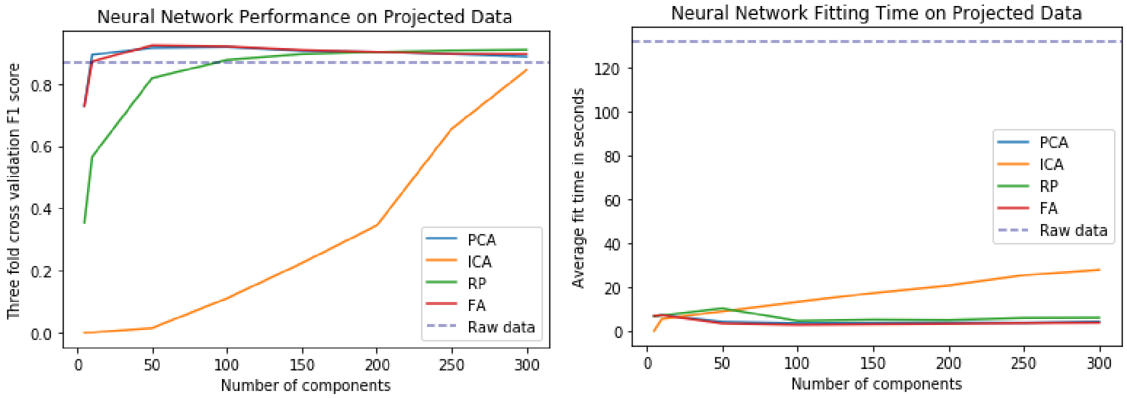


Figure 15 - Applying Neural Network on Projected and Raw Data for Dataset 1

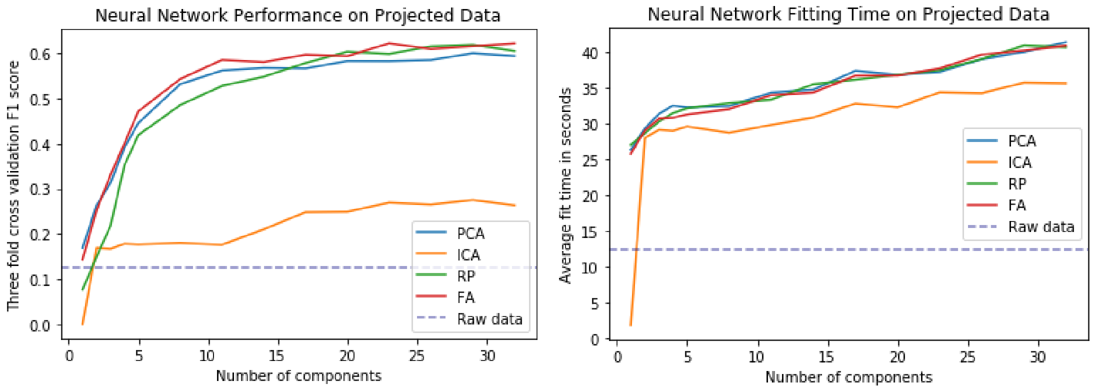


Figure 16 - Applying Neural Network on Projected and Raw Data for Dataset 2

Part 5. Neural network learner on dimensionality reduced and clustered datasets

In this part, I first applied dimension reduction on the original dataset, then clustered the projected data, and finally fitted neural network on the clustering transformed data which is a cluster-distance matrix for k-means, and a cluster-probability matrix for EM.

As clustering can also be understood as a dimension reduction method, the cross validation F1 scores achieved by k-means and EM projection are less than half of the F1 scores without clustering projection. The accuracy obtained by clustering shows the alignment between clustering and classification which is similar to the discussion in Part 1. And similar to Part 1, NN after PCA and clustering has the highest performance at small numbers of components, while NN after ICA/RP and clustering has increasing performance as the number of components increases.

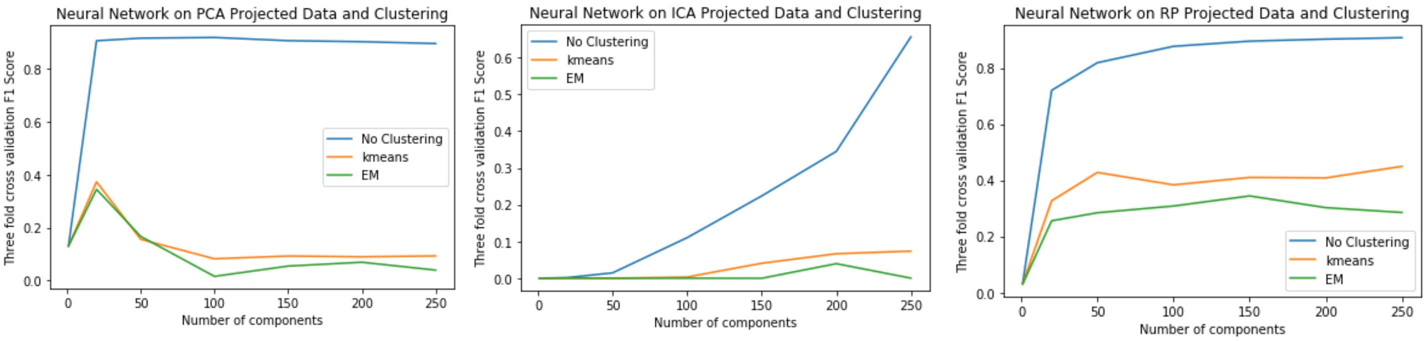


Figure 17 - Applying Neural Network on Projected and Clustered data for Dataset 1

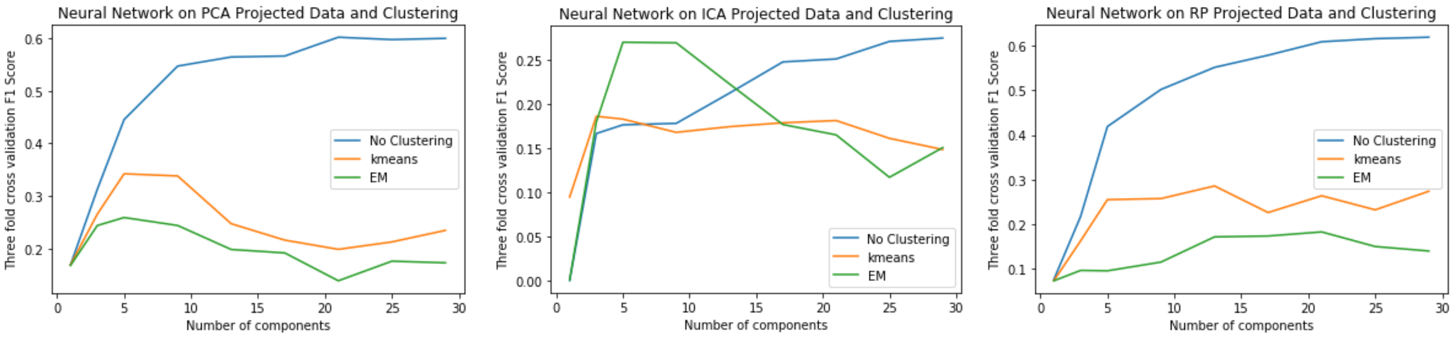


Figure 18 - Applying Neural Network on Projected and Clustered data for Dataset 2

Conclusion

Dimensionality reduction algorithms specifically PCA, ICA, RP, and FA that were studied in this paper are considered as feature extraction methods, and are popular preprocessing steps before unsupervised or supervised learning. From experiments on the two datasets, projecting data with PCA/RP/FA can generally yield comparable learning performance comparing with learning on the original data, with benefits of reduced dimensions and the resulting computing and storing efficiency. And if the original data is noisy, they can achieve even better performance due to the reduction of noise and correlations. ICA is more suitable for problems such as separating mixed signals, or detecting "meaningful" components, but is not good at retaining information with smaller dimensions for later learning steps. From data perspective, datasets such as the digit pixel data which has high dimensions but much redundancy can be projected into much lower dimensional space with small information loss.

This study also tried to shed some light upon the connection between clustering and classification. Clustering assignment and class labels have some degree of alignment in the experiments, but not full alignment mainly because the unsupervised clustering have no build-in feature selection, feature weight learning, or complex feature transformation as the supervised classification. The connection between clustering and PCA is also observed in the study.

References

Ding, C., & He, X. (2004, July). K-means clustering via principal component analysis. In Proceedings of the twenty-first international conference on Machine learning (p. 29). ACM.

Hyvärinen, & Oja. (2000). Independent component analysis: Algorithms and applications. Neural Networks, 13(4), 411-430.

Cohen, M. B., Elder, S., Musco, C., Musco, C., & Persu, M. (2015, June). Dimensionality reduction for k-means clustering and low rank approximation. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing (pp. 163-172). ACM.

Boutsidis, C., Zouzias, A., & Drineas, P. (2010). Random projections for k-means clustering. In Advances in Neural Information Processing Systems (pp. 298-306).

Hinton, G., & Salakhutdinov, R. (2006). Reducing the Dimensionality of Data with Neural Networks. Science, 313(5786), 504-507.

LeCun, Y. A., Bottou, L., Orr, G. B., & Müller, K. R. (2012). Efficient backprop. In Neural networks: Tricks of the trade (pp. 9-48). Springer, Berlin, Heidelberg.

Bingham, E., & Mannila, H. (2001, August). Random projection in dimensionality reduction: applications to image and text data. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 245-250). ACM.